

 Stagnant Standards Track: Core

EIP-2474: Coinbase calls

Authors	Ricardo Guilherme Schmidt (@3esmit)
Created	2020-01-19
Discussion Link	https://ethresear.ch/t/gas-abstraction-non-signed-block-validator-only-procedures/4388/2

Table of Contents

- Simple Summary
- Abstract
- Motivation
- Specification
- Rationale
- Backwards Compatibility
- Test Cases
- Implementation
- Security Considerations
- Copyright

Simple Summary

Allow contracts to be called directly by `block.coinbase` (block validator), without a transaction.

Abstract

In proof-of-work blockchains, validators are known as miners.

The validator might want to execute functions directly, without having to sign a transaction. Some examples might be presenting a proof in a contract for an change which also benefits the validator.

A notable example would be when a validator want to act as an EIP-1077 Gas Relay, incentivized to pick up fees from meta transactions. Without this change, they can do so by signing from any address a `gasPrice = 0` transaction with the gas relayed call. However this brings an overhead of a signed transaction by validator that does nothing, as `msg.sender` is never used, and there is no gas cost to EVM charge.

This proposal makes possible to remove this unused ecrecover.

Motivation

In order to reduce the overhead of calls that don't use `msg.sender` and are being called by validator with `tx.gasPrice = 0`.

Specification

The calls to be executed by `block.coinbase` would be included first at block, and would consume normally the gas of block, however they won't pay/cost gas, instead the call logic would pay the validator in other form.

Would be valid to execute any calls without a transaction by the block coinbase, except when the validator call tries to read `msg.sender`, which would throw an invalid jump.

Calls included by the validator would have `tx.origin = block.coinbase` and `gas.price = 0` for the rest of call stack, the rest follows as normal calls.

Rationale

TBD

Backwards Compatibility

`tx.origin = block.coinbase` could cause some issues on bad designed contracts, such as using `tx.origin` to validate a signature, an analysis on how contracts use `tx.origin` might be useful to decide if this is a good choice.

Test Cases

TBD

Implementation

TBD

Security Considerations

TBD

Copyright

Copyright and related rights waived via CC0.

Citation

Please cite this document as:

Ricardo Guilherme Schmidt (@3esmit), "EIP-2474: Coinbase calls [DRAFT]," *Ethereum Improvement Proposals*, no. 2474, January 2020. [Online serial]. Available: <https://eips.ethereum.org/EIPS/eip-2474>.